UNIVERSITAS INDONESIA

DETEKSI OBJEK BERGERAK DENGAN ALGORITMA *BACKGROUND SUBTRACTION* BERBASIS *GAUSSIAN MIXTURE MODEL*

LAPORAN PROYEK

KELOMPOK 6

Elyaser Ben Guno	**1806195135**
Thara Adiva Putri Candra	**1806187594**

DEPARTEMEN TEKNIK ELEKTRO
FAKULTAS TEKNIK
DEPOK

MEI 2021

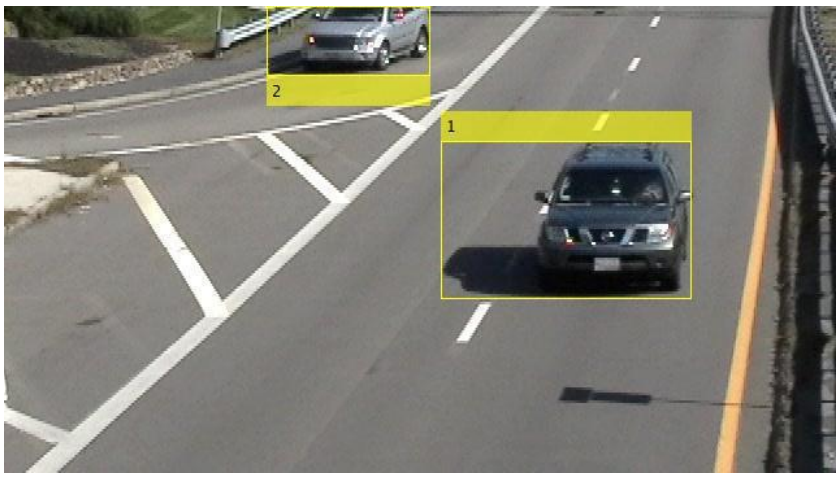
DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Deteksi objek bergerak merupakan salah satu teknik yang digunakan dalam *computer vision* dan pemrosesan gambar. Teknik ini digunakan untuk menentukan apakah ada objek bergerak yang terdeteksi. Pengaplikasian dari deteksi objek bergerak ini contohnya untuk *video surveillance*, pengenalan aktivitas, pemantauan kondisi jalanan, keamanan bandara, pemantauan perlindungan di sepanjang perbatasan laut dan masih banyak lagi.



Gambar 1.1 Deteksi Objek Bergerak

Deteksi objek bergerak bekerja dengan mengenali pergerakan fisik suatu objek di suatu tempat atau wilayah tertentu, dengan cara melakukan segmentasi antara objek bergerak (*foreground*) dari area / wilayah yang diam (*background*). Di antara semua metode deteksi objek bergerak, metode tersebut dapat dikategorikan menjadi 4, yaitu:

1. *Background Subtraction.*
2. *Frame Differencing*.
3. *Temporal Differencing*.
4. *Optical flow*.

Adapun yang akan dibahas pada proyek ini yaitu metode *background subtraction* berbasis *gaussian mixture model*. Alasan kami mengambil metode *background subtraction* sebagai metode kami, yaitu karena metode tersebut merupakan metode yang umum digunakan untuk mendeteksi objek bergerak dari kamera yang sifatnya stasioner.

1.2 Tujuan

Tujuan dari penulisan laporan proyek ini yaitu untuk memenuhi nilai tugas mata kuliah pengolahan sinyal dan layanan multimedia. Selain itu, tujuan dari proyek ini sendiri yaitu untuk membuat dan mensimulasikan deteksi objek bergerak dengan algoritma *background subtraction* berbasis *gaussian mixture model* menggunakan perangkat lunak Matlab. Dari simulasi yang dilakukan, diharapkan dapat diketahui beberapa hal seperti:

a. Bagaimana cara kerja algoritma *background subtraction* berbasis *gaussian mixture model* dalam mendeteksi objek bergerak.
b. Pengaruh parameter-parameter yang ada pada *gaussian mixture model* terhadap hasil deteksi dan juga waktu komputasi.

1.3 Batasan Masalah

Beberapa hal yang menjadi batasan masalah pada proyek ini diantaranya adalah sebagai berikut ini.

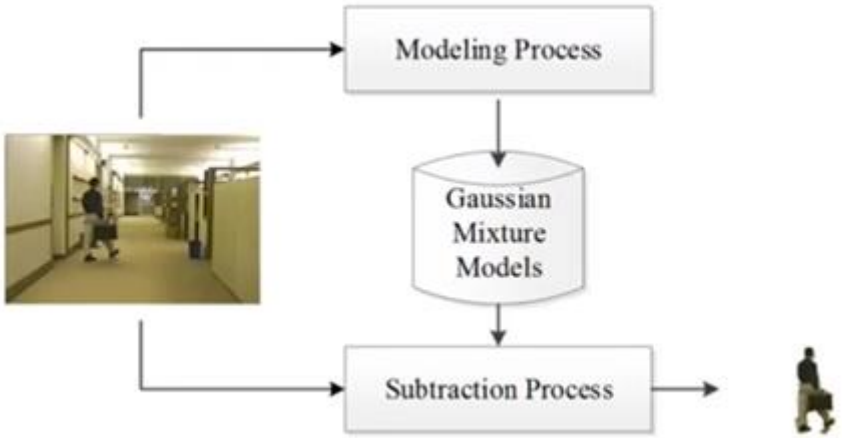
a. Deteksi tidak dilakukan secara *real time* karena keterbatasan kekuatan komputasi dari perangkat keras yang digunakan.
b. Video uji coba yang digunakan merupakan video yang kameranya bersifat stasioner (tidak bergerak).

BAB II

LANDASAN TEORI

2.1 Algoritma *Background Subtraction* Berbasis GMM (*Gaussian Mixture Model*)

Background subtraction merupakan metode umum yang digunakan untuk mendeteksi objek bergerak. Dalam proyek ini, digunakan algoritma *background subtraction* berbasis GMM (*Gaussian Mixture Model*). Penggunaan GMM ini bertujuan untuk mensegmentasi *region of interests* yakni *foreground* dari *background processes*. Berikut adalah *framework* dari algoritma *background subtraction* berbasis GMM.



Gambar 2.1 *Framework* Algoritma *Background Subtraction* Berbasis GMM

Modeling process bertujuan untuk membentuk GMM yang mendeskripsikan *scene*. *Modeling process* akan memperbarui parameter dari model untuk beradaptasi sesuai *scene*. Lalu, pada *subtraction process* terjadi substraksi *background* dari gambar yang saat ini terobservasi untuk mendapatkan *foreground mask*. Model GMM ini dalam mendeteksi objek bergerak mempertimbangkan perubahan intensitas pencahayaan dan perubahan penampilan (misal, perpindahan objek dari satu tempat ke tempat lain). Dalam prosesnya, algoritma ini menjalankan suatu pemodelan yang dinamakan *on-line mixture model*.

2.1.1 *On-line Mixture Model*

Nilai dari suatu piksel terhadap waktu disebut sebagai "*pixel process*", yang mana merupakan deret waktu dari nilai piksel, contohnya seperti nilai skalar dari *gray values* atau vektor dari gambar berwarna. Pada tiap waktu *t*, *history* dari tiap piksel tertentu $\{x_0, y_0\}$ dapat dirumuskan sebagai berikut ini.

$$\{X_1, ..., X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\}$$

Dimana I merupakan *image sequence*. Sebagaimana yang disebutkan sebelumnya, model GMM ini mempertimbangkan perubahan intensitas pencahayaan dan perubahan tampilan. Jika suatu cahaya berubah pada *scene* yang statis, maka perubahan tersebut dapat dimodelkan dengan distribusi gaussian untuk melacak perubahan-perubahan yang terjadi pada *scene* tersebut.

Aspek dari variasi piksel terjadi jika terdapat objek bergerak di dalam *scene*. *History* terkini dari tiap pixel $\{X_1, \ldots, Xt\}$ dapat dimodelkan dengan *mixture* (campuran) dari sejumlah K distribusi Gaussian. Probabilitas dari observasi nilai piksel saat ini yaitu.

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t})$$

Dimana K merupakan banyaknya distribusi gaussian. $\omega_{i,t}$ merupakan estimasi bobot dari gaussian ke-i di dalam *mixture* pada waktu t. $\mu_{i,t}$ merupakan nilai rata-rata dari gaussian ke-i di dalam *mixture* pada waktu t. $\sum_{i,t}$ merupakan matriks kovarians dari gaussian ke-i di dalam *mixture* pada waktu t. η merupakan fungsi kepadatan probabilitas gaussian.

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu_t)^T \Sigma^{-1} (X_t - \mu_t)}$$

K ditentukan berdasarkan memori yang tersedia dan juga berdasarkan kekuatan komputasi. Nilai K yang biasa digunakan adalah 3, 4 dan 5. Lalu, untuk alasan komputasi, matriks kovarians diasumsikan berbentuk seperti berikut.

$$\Sigma_{k,t} = \sigma_k^2 \mathbf{I}$$

Diasumsikan bahwa nilai piksel merah, hijau dan biru masing-masing bersifat independen dan memiliki varians yang sama. Asumsi ini memungkinkan untuk menghindari perhitungan inversi matriks yang memakan kekuatan komputasi dan sebagai gantinya, hal tersebut dapat mengurangi akurasi.

Distribusi dari nilai tiap piksel yang terobservasi ditandai oleh campuran (*mixture*) distribusi gaussian. Nilai piksel baru akan direpresentasikan oleh salah satu komponen utama dari model *mixture* dan digunakan untuk memperbarui model.

Setiap nilai piksel baru X_t, dicek terhadap distribusi K gaussian yang telah ada, sampai ditemukan *match*. *Match* didefinisikan sebagai nilai piksel dalam 2,5 standar deviasi pada distribusi. Jika tidak ditemukan distribusi K gaussian yang cocok (*match*) dengan nilai piksel sekarang, distribusi yang kemungkinannya paling kecil diganti dengan distribusi baru, dengan nilai saat ini sebagai rata-ratanya, nilai varians awal yang tinggi dan bobot prior yang rendah. Bobot prior dari distribusi K pada waktu t, $\omega_{k,t}$ disesuaikan sebagai berikut.

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t})$$

Dimana α merupakan *learning rate*, dan $M_{k,t}$ bernilai 1 untuk model yang *matched* dan 0 untuk model sisanya. Setelah aproksimasi ini, bobot dinormalisasi kembali. 1/ α merupakan *time constant* yang menentukan kecepatan perubahan parameter distribusi. Parameter μ dan σ untuk parameter *unmatched* tetap sama. Parameter dari distribusi yang *match* dengan observasi baru akan diperbarui sebagai berikut.

$$\mu_t = (1-\rho)\mu_{t-1} + \rho X_t$$
$$\sigma_t^2 = (1-\rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t)$$
$$\rho = \alpha\eta(X_t|\mu_k, \sigma_k)$$

Dimana ρ merupakan nilai *learning rate* kedua. Kelebihan dari metode ini adalah ketika sesuatu menjadi bagian dari latar belakang (*background*), maka hal tersebut tidak menghancurkan model *background* yang telah ada. Warna dari *background* asli tetap di dalam *mixture* sampai ia menjadi K^{th} yang paling memungkinkan dan ketika ada warna baru yang terobservasi. Selain itu, jika suatu objek diam untuk waktu yang lama lalu objek tersebut bergerak, maka distribusi yang mendeskripsikan *background* sebelumnya tetap ada dengan nilai μ dan σ^2 yang sama, namun nilai ω menjadi lebih kecil dan secara cepat bergabung kembali dengan *background*.

2.1.2 Estimasi Model *Background*

Dengan berubahnya parameter model *mixture* dari tiap piksel, dapat ditentukan gaussian mana dari *mixture* yang kemungkinan besar diproduksi oleh *background processes*. Dengan begitu, dibutuhkan metode untuk menentukan berapa porsi model *mixture* yang dengan baik merepresentasikan *background processes*. Pertama, Gaussian diurutkan berdasar nilai ω/σ. Setelah mengestimasi ulang parameter dari *mixture*, lalu menyortir dari distribusi yang *matched* ke distribusi *background* yang paling memungkinkan Hal tersebut dilakukan karena hanya nilai relatif model *matched* yang akan berubah. Pengurutan dari model ini secara efektif mengurutkan distribusi *background* yang paling memungkinkan tetap berada di atas dan distribusi *transient background* yang kemungkinannya kecil akan turun ke bawah dan digantikan oleh distribusi baru. Lalu, distribusi B pertama dipilih sebagai model *background* dimana B bernilai sebagai berikut.

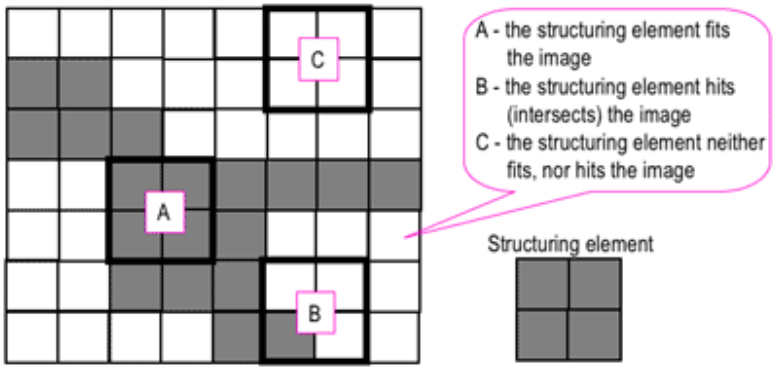
$$B = argmin_b \left(\sum_{k=1}^{b} \omega_k > T \right)$$

Dimana T merupakan ukuran dari porsi minimum dari data yang ditetapkan oleh *background*. Jika nilai T yang dipilih bernilai kecil, model *background* akan bersifat unimodal. Jika nilai T besar, maka akan terjadi distribusi multimodal akibat adanya gerakan repetitif dari *background* (contohnya seperti gerakan daun pada pohon, bendera yang tertiup angin dan lain-lain).

2.2 Operasi Morfologi

Setelah melewati proses *background subtraction* berbasis GMM, selanjutnya dilakukan pemfilteran dengan operasi morfologi. Pemrosesan gambar secara morfologi merupakan kumpulan dari operasi non-linier yang berhubungan dengan bentuk atau morfologi dari fitur yang ada pada gambar. Operasi ini dapat diterapkan pada gambar *grayscale*. Operasi morfologi hanya mengandalkan pengurutan relatif dari nilai piksel, bukan pada nilai numeriknya, dan oleh karena itu sangat cocok untuk pemrosesan gambar biner.

Operasi morfologi menyelidiki gambar berdasarkan bentuk atau templat kecil yang disebut sebagai elemen penataan (*structuring element*). Elemen penataan diposisikan di semua lokasi yang memungkinkan dalam gambar dan dibandingkan dengan lingkungan piksel yang sesuai.



Gambar 2.2 Menyelidiki Gambar Biner dengan Elemen Penataan

Dalam operasi morfologi, terdapat beberapa operasi fundamental seperti erosi dan pelebaran (*dilation*), operasi *compound* seperti *opening* dan *closing*. Berikut ini penjelasan dari operasi fundamental tersebut.

2.2.1 Erosi

Erosi dari gambar biner *f* dengan elemen penataan *s* akan menghasilkan gambar biner baru yang direpresentasikan oleh *g*, dimana *g* memiliki persamaan sebagai berikut.

$$g = f \ominus s$$

Erosi menghilangkan detail skala kecil dari gambar biner dan secara simultan mereduksi ukuran dari *region of interest*. Berikut ini merupakan efek dari operasi erosi terhadap gambar biner.

Gambar 2.3 Efek Erosi terhadap Gambar Biner

2.2.2 Pelebaran

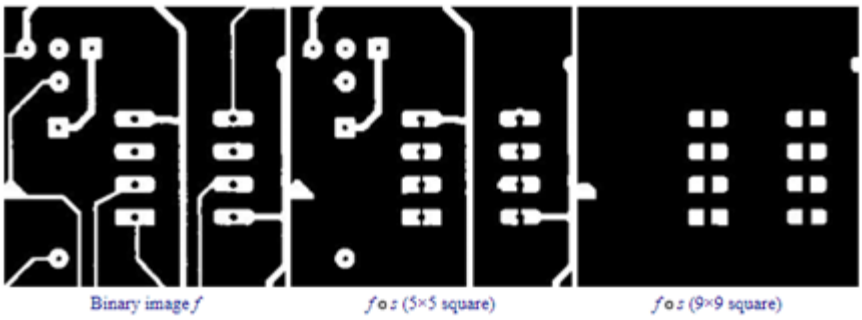
Pelebaran atau *dilation* dari gambar disimbolkan dengan persamaan $f \oplus s$. Operasi pelebaran ini akan memiliki efek berkebalikan dari erosi dimana akan menambah lapisan piksel ke dalam dan luar dari batas *region of interest*. Berikut ini merupakan hasil dari operasi pelebaran.

Gambar 2.4 Efek Pelebaran terhadap Gambar Biner

2.2.3 *Opening*

Operasi *opening* didapat dengan terlebih dahulu menerapkan erosi pada gambar lalu melebarkannya. *Opening* menghilangkan koneksi sempit dan garis di antara 2 area. Berikut ini merupakan persamaan dari operasi *opening* beserta efek dari operasi *opening* terhadap gambar biner.

$$f \circ s = (f \ominus s) \oplus s$$



Gambar 2.5 Efek Operasi *Opening* terhadap Gambar Biner

2.2.4 *Closing*

Operasi *closing* didapat dengan terlebih dahulu melebarkan gambar lalu mengerosikannya. Urutan operasi tersebut merupakan kebalikan dari *opening*. *Closing* akan mengisi area hitam sempit atau lubang pada gambar. Berikut ini merupakan persamaan dari operasi *closing* beserta efek dari operasi *closing* terhadap gambar biner.

$$f \bullet s = (f \oplus s) \ominus s$$

Gambar 2.6 Efek Operasi *Closing* terhadap Gambar Biner

2.3 *Blob Detection*

Dalam *computer vision*, metode deteksi blob ditujukan untuk mendeteksi wilayah dalam citra digital yang sifatnya berbeda, seperti kecerahan atau warna, yang akan dibandingkan dengan wilayah sekitarnya. Secara informal, blob adalah wilayah gambar yang beberapa propertinya konstan atau mendekati konstan, semua titik di dalam blob dapat dianggap mirip satu sama lain. Dalam proyek ini, *blob detection* berguna untuk mendeteksi wilayah dalam citra yang merepresentasikan objek bergerak. Dari proses *blob detection* ini akan didapatkan data koordinat sentroid blob yang terdeteksi dan koordinat dari *bounding box*. Koordinat tersebut dibutuhkan untuk melakukan proses *tracking* menggunakan Kalman Filter

2.4 Kalman Filter

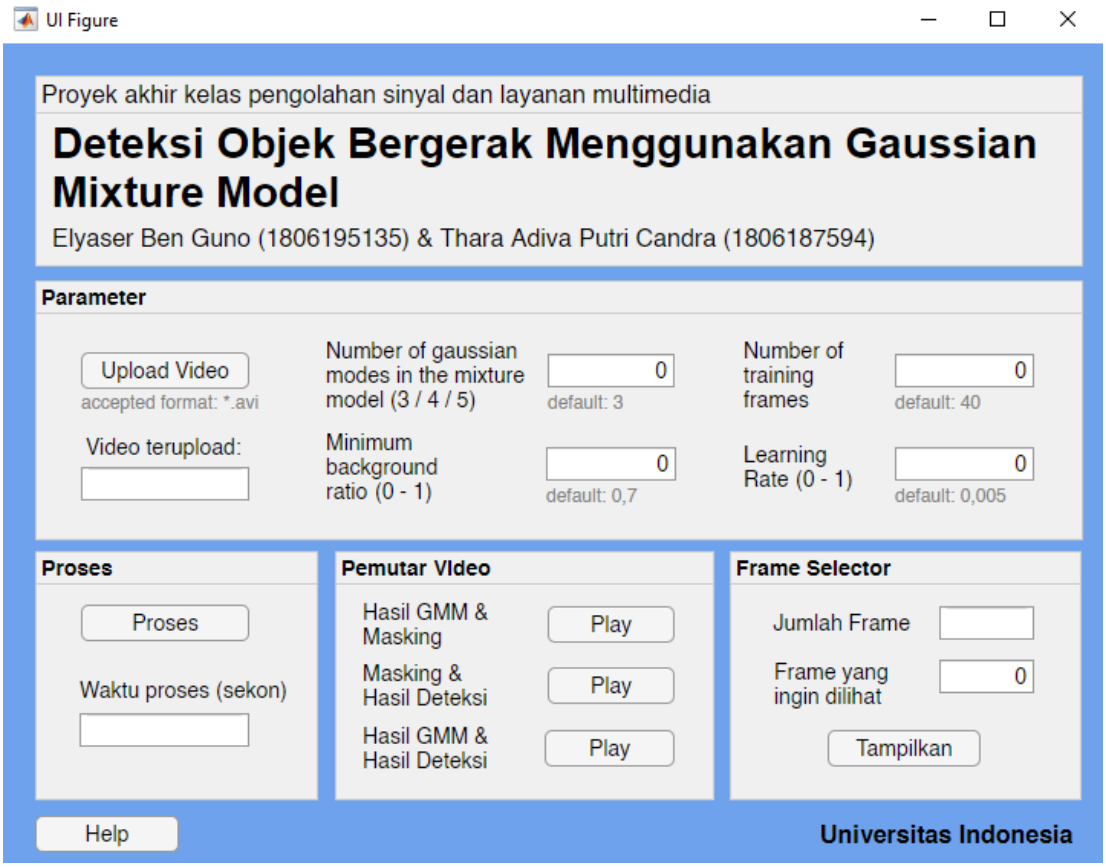
Kalman filter memiliki banyak kegunaan, diantaranya dapat diaplikasikan di bidang kontrol, navigasi, *computer vision* dan ekonometrika. Dalam proyek ini, setelah objek dideteksi dengan *blob detection*, selanjutnya digunakan kalman filter untuk melacak objek bergerak tersebut. Dimana pelacakan ini dapat melacak ketika ada objek bergerak baru yang muncul pada *scene*, lalu ketika objek tersebut menghilang dari *scene* maka pelacakan dapat berakhir. Pelacakan dilakukan dengan mengestimasi pelacakan dari *frame* sebelumnya dan setiap saat dilakukan *update*.

BAB III

RANCANGAN SIMULASI

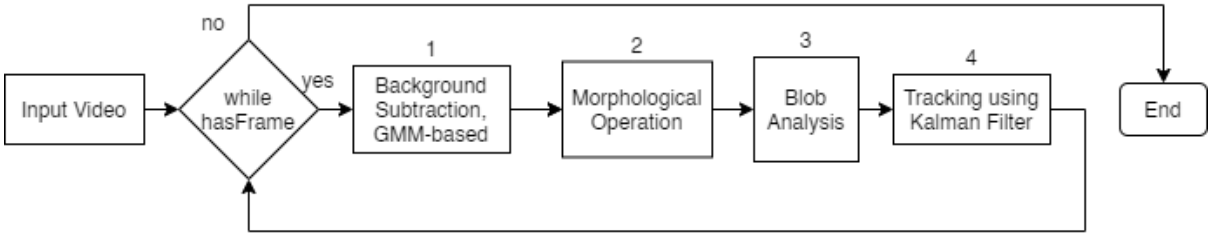
3.1 Metode/Rancangan Simulasi

Simulasi dari proyek ini dirancang dengan bentuk aplikasi yaitu dengan *Graphical User Interface* (GUI) sehingga memudahkan *user* dalam menggunakannya dan lebih *user friendly*. Simulasi ini dijalankan di aplikasi MATLAB (minimal MATLAB 2018b), dan memerlukan *image processing toolbox* dan *computer vision toolbox* yang perlu di-*install* di aplikasi MATLAB tersebut. Terdapat dua file MATLAB yang digunakan untuk menjalankan simulasi ini, yaitu file "MotionObjectDetector.mlapp" sebagai aplikasi utama dalam bentuk GUI, dan "DOBmGMM.m'' sebagai program utama untuk menjalankan simulasi deteksi objek bergerak menggunakan *Gaussian Mixture Model* ini. Kode lengkap dari kedua file tersebut tercantum pada lampiran. Berikut ini merupakan tampilan aplikasi deteksi objek bergerak yang kami buat.



Gambar 3.1 Aplikasi Deteksi Objek Bergerak

Sebelum menjalankan simulasi, *user* perlu menyiapkan sebuah video (dalam format *.avi) yang akan dijadikan bahan untuk mendeteksi objek bergerak yang ada pada video tersebut. Lalu, *user* menjalankan aplikasi utama "MotionObjectDetector.mlapp". *User* mengunggah video yang akan digunakan, lalu *user* juga harus mengisi beberapa parameter seperti jumlah *gaussian mixture model*, *minimum background ratio*, *learning rate* dan jumlah *training frames* terlebih dahulu pada kolom yang disediakan. Setelah itu, tekan tombol proses dan secara otomatis akan menjalankan program "DOBmGMM.m'' untuk mendeteksi objek bergerak. Berikut ini merupakan algoritma dari program deteksi objek bergerak yang kami buat.



Gambar 3.2 Algoritma Program Deteksi Objek Bergerak

Diagram diatas menggambarkan bagaimana program ini berjalan untuk memproses suatu video. Dalam simulasi ini, yang menjadi input adalah sebuah video yang diunggah oleh *user*. Pada program MATLAB, terdapat suatu fungsi yaitu "while hasFrame", dimana program akan berjalan selama masih terdapat *frame* pada video tersebut, dan nilai *frame* akan di-*increment* sampai sudah tidak terdapat *frame* yang dapat diproses, sehingga program akan otomatis berhenti. Dalam program ini, terdapat empat proses yang berjalan, yaitu (1) *Background Subtraction* berbasis *Gaussian Mixture Model*, (2) Operasi Morfologi, (3) Analisa Blob dan (4) *Tracking* menggunakan Kalman Filter. Pada simulasi ini, proyek ini menekankan pada pembelajaran proses 1, yaitu *Background Subtraction* berbasis *Gaussian Mixture Model.* Maka dari itu, pada simulasi ini, parameter-parameter yang akan diubah dan diamati pengaruhnya terhadap hasil adalah parameter dari proses 1.

Setelah proses selesai, dihasilkan 3 video antara lain video *masking* hasil *background subtraction* berbasis GMM, video *masking* setelah melewati proses operasi morfologi, *blob analysis* dan *tracking* dengan kalman filter dan video hasil akhir deteksi objek bergerak. *User*

hanya bisa menampilkan hanya 2 video pada saat bersamaan. Selain itu, terdapat juga *frame selector* untuk melihat *frame* tertentu dari video asli beserta 3 video yang dihasilkan dari proses agar memudahkan *user* untuk menganalisis perbedaan video yang dihasilkan dari tiap prosesnya.

3.2 Variasi & Ketentuan Simulasi

Untuk pengujian simulasi proyek ini, terdapat beberapa variasi yang bisa dilakukan dengan mengubah nilai dari parameter-parameternya, seperti jumlah *gaussian mixture*, *learning rate* beserta porsi minimum *background*. Akan dianalisis hubungan antara jumlah *gaussian mixture* dengan waktu komputasi. Begitu pun juga dengan *learning rate*, akan dianalisis hubungannya dengan waktu komputasi. Dari dua variasi simulasi tersebut dapat diketahui bagaimana jumlah distribusi gaussian mempengaruhi waktu komputasi, serta berapa *learning rate* yang paling efisien untuk digunakan dalam pemodelan. Porsi minimum *background* juga dapat divariasikan untuk mengetahui perbedaan hasil deteksi nya.

Simulasi dijalankan dengan detail perangkat keras sebagai berikut, prosesor Intel(R) Core (TM) i5-3320M CPU @ 2.60 GHz (4 CPUs) ~2,6 GHz; memori 8192MB RAM dan kartu grafis Intel(R) HD Graphics 4000. Simulasi dijalankan pada kondisi tidak ada program lain yang berjalan dan tidak terkoneksi ke internet.

Pada simulasi variasi jumlah *gaussian mixture* dan *learning rate*, masing-masing simulasi dilakukan 3 set percobaan untuk memvalidasi hubungan perubahan parameter terhadap waktu komputasi. Tiap set percobaan, selain parameter yang divariasikan, parameter lainnya akan di-*set* nilainya sesuai *default*-nya.

BAB IV

SIMULASI DAN ANALISIS

4.1 Simulasi Deteksi Objek Bergerak

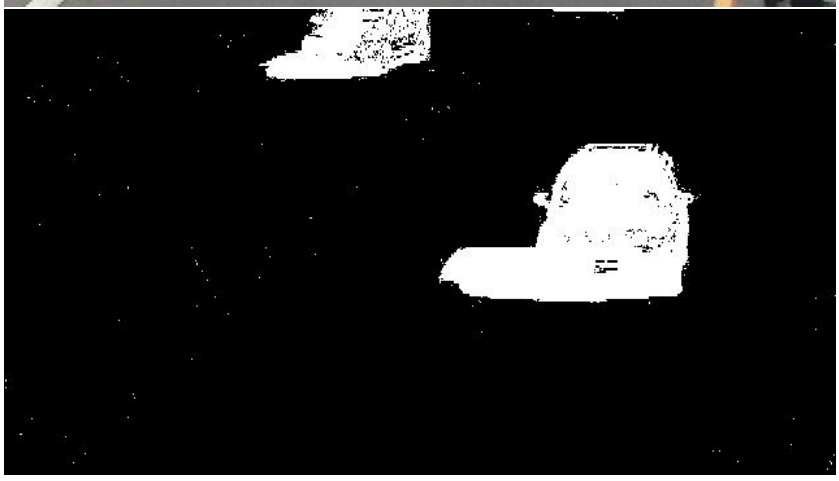
Pada simulasi ini, dilakukan percobaan dengan parameter *default* sebagai berikut.

a. Jumlah K Gaussian *mixture* : 3
b. *Learning rate*, α : 0,005
c. Porsi min. *Background*, T : 0,7
d. Jumlah frame untuk *train* : 40

Dari percobaan yang telah dilakukan, didapat hasil deteksi objek bergerak sebagaimana pada gambar berikut ini:

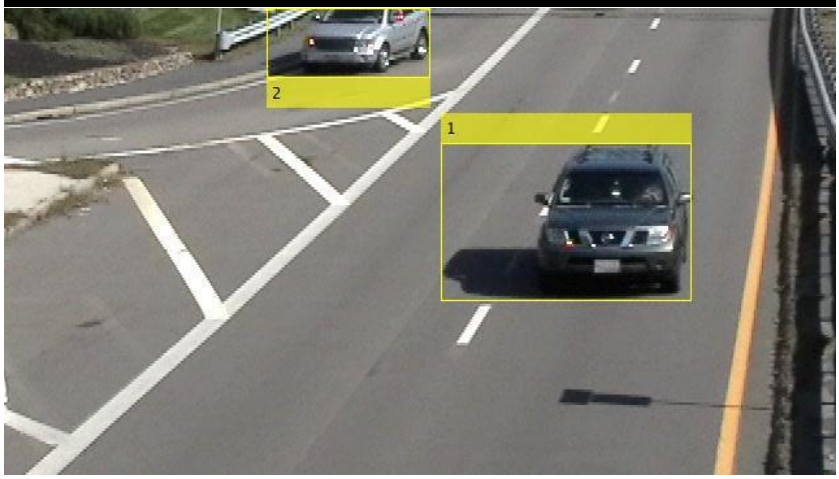
a.

b.



c.

d.



Gambar 4.1 Tangkapan Layar pada Frame ke-153, (a) video asli, (b) *masking* hasil dari *foreground detector* dengan GMM, (c) *masking* setelah melalui operasi morfologi, *blob analysis* & *tracking*, (d) video hasil deteksi

Dari Gambar 4.1 (b), diketahui bahwa setelah melalui proses deteksi objek bergerak masih terdapat piksel-piksel yang tidak merepresentasikan objek bergerak (*noise*). Dengan begitu, maka diperlukan operasi morfologi untuk menghilangkan *noise* tersebut. Hasil dari operasi tersebut ditampilkan pada Gambar 4.1 (c) dimana terlihat hasil deteksi lebih 'bersih' dan pada gambar tersebut sudah ditambahkan *bounding box* serta hasil *tracking*. Sedangkan, pada gambar 4.1. (d) merupakan hasil akhir dari deteksi objek bergerak. Terlihat bahwa kendaraan (objek bergerak) yang muncul di *frame* akan dihitung jumlahnya dan dilacak jumlahnya. Jika kendaraan sudah lewat (sudah hilang dari *frame*), maka *track* akan dihapus. Jika terdapat kendaraan baru yang memasuki *frame*. maka akan dibuat *track* baru dan kendaraan tersebut memiliki angka baru.

4.2 Simulasi Variasi Jumlah GMM

Pada simulasi ini jumlah GMM atau *Gaussian Mixture Model* divariasikan sejumlah 3, 4 dan 5 untuk mengetahui efek variasi jumlah GMM terhadap waktu komputasi. Sedangkan, untuk parameter lainnya dibuat tetap untuk setiap set percobaan seperti pada Bagian 4.1, yaitu:

a. *Learning rate*, α : 0,005
b. Porsi min. *Background*, T : 0,7
c. Jumlah frame untuk *train* : 40

Dari simulasi yang dilakukan, didapat hasil waktu komputasi sebagai berikut.

Tabel 4.1 Efek Variasi Jumlah GMM terhadap Waktu Komputasi

Jumlah GMM	Waktu Komputasi (s)		
	Set Percobaan 1	Set Percobaan 2	Set Percobaan 3
3	22,0593	22,0742	21,8948
4	22,4316	22,5030	22,2087
5	22,6695	22,5363	22,6126

Dapat dilihat pada Tabel 4.1 bahwa ketika jumlah GMM yang digunakan semakin banyak, maka waktu komputasinya akan semakin lama. Hal ini didasarkan dari semakin banyak jumlah GMM yang digunakan, maka semakin banyak distribusi Gaussian yang harus dihitung. Sedangkan, dari hasil deteksinya tidak menunjukkan perbedaan yang signifikan.

4.3 Simulasi Variasi *Learning Rate*

Pada simulasi ini nilai *learning rate* akan divariasikan menjadi 0,00005; 0,005 dan 0,5 untuk mengetahui efeknya terhadap waktu komputasi. Sedangkan, untuk parameter lainnya dibuat tetap untuk setiap set percobaan seperti pada Bagian 4.1, yaitu:

a. Jumlah K Gaussian *mixture* : 3
b. Porsi min. *Background*, T : 0,7
c. Jumlah *frame* untuk *train* : 40

Berikut ini hasil dari percobaan tersebut.

Tabel 4.2 Efek Variasi Nilai *Learning Rate* terhadap Waktu Komputasi

Nilai *Learning Rate*	Waktu Komputasi (s)		
	Set Percobaan 1	Set Percobaan 2	Set Percobaan 3
0,00005	22,4655	22,8928	22,3489
0,005	22,5788	23,0712	23,1053
0,5	22,8815	24,9994	25,7331

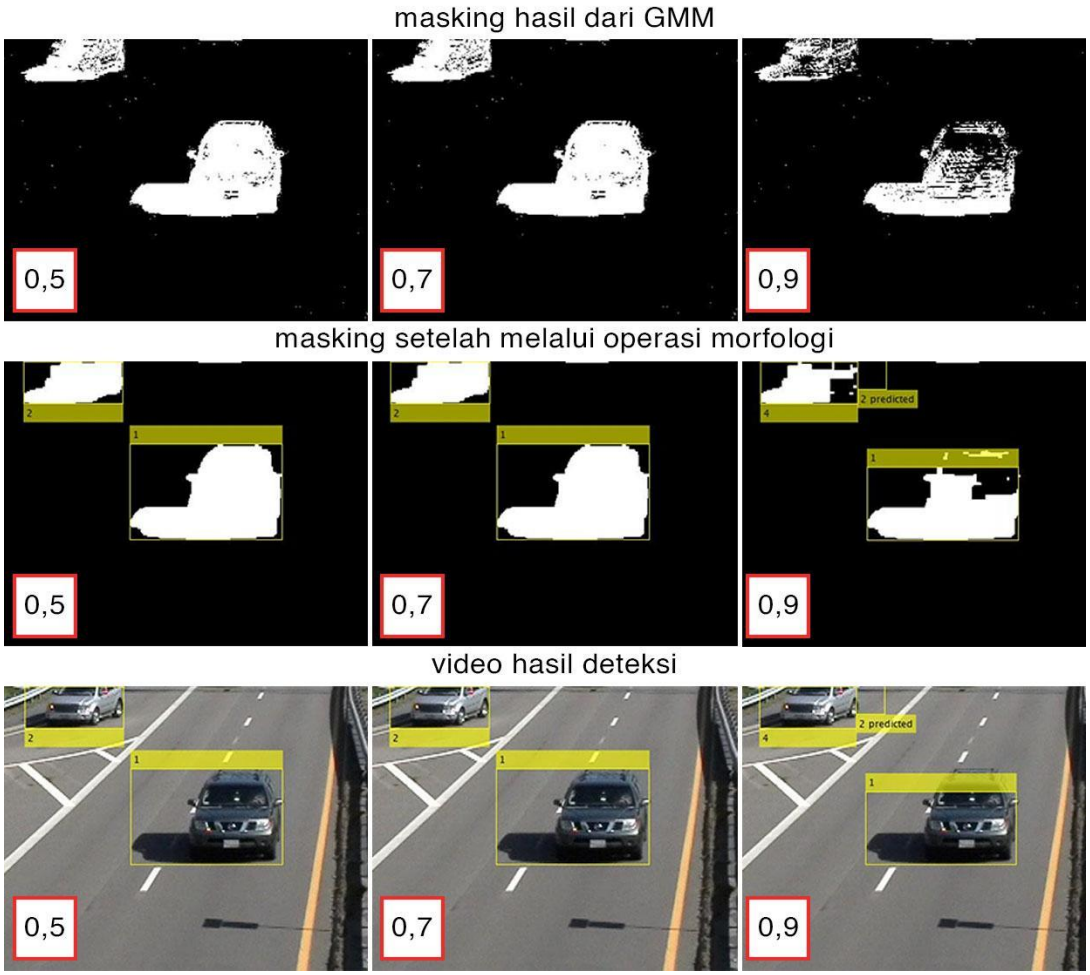
Dapat dilihat pada Tabel 4.2 bahwa ketika nilai *learning rate* semakin meningkat maka waktu komputasinya juga akan ikut meningkat. Tapi bukan berarti akan terjadi hal yang sama pada model lainnya. Pada kasus ini, *learning rate* yang besar akan menyebabkan bertambahnya varians pada waktu *training* sehingga waktu komputasinya bertambah. Namun, perlu diperhatikan pula, apabila *learning rate* sangat kecil, proses *update* akan memakan waktu yang lama akibat langkah / *step* yang diambil sangat kecil. Dengan begitu, hal tersebut juga akan membuat bertambahnya waktu komputasi. Dapat disimpulkan bahwa dari ketiga nilai *learning rate* yang diujikan pada kasus ini, nilai yang paling efektif adalah 0,00005.

4.4 Simulasi Variasi Porsi Minimum *Background*

Pada simulasi ini nilai porsi minimum *background* (T) divariasikan menjadi 0,5; 0,7 dan 0,9 untuk mengetahui efeknya terhadap deteksi yang dihasilkan. Sedangkan, untuk parameter lainnya dibuat tetap seperti pada Bagian 4.1, yaitu:

a. Jumlah K Gaussian *mixture* : 3
b. *Learning rate*, α : 0,005
c. Jumlah frame untuk *train* : 40

Berikut ini hasil dari percobaan tersebut.



Gambar 4.2 Efek Variasi Nilai Porsi Minimum *Background* terhadap Hasil Deteksi

Dari Gambar 4.2 dapat dilihat bahwa perbedaan hasil deteksi antara nilai porsi minimum *background* 0,5 dan 0,7 tidak terlalu signifikan. Sedangkan, jika dibandingkan dengan nilai 0,9 maka terdapat perbedaan hasil deteksi yang signifikan. Pada nilai 0,9 didapat hasil deteksi yang ‘kasar’ dan tidak menyerupai objek bergerak yang kita ingin deteksi. Ditinjau dari *masking* hasil GMM-nya, terlihat bahwa terdapat ‘lubang’ yang ditandai adanya piksel berwarna hitam pada area yang seharusnya dideteksi sebagai objek bergerak.

BAB V

PENUTUP

5.1 Kesimpulan

Setelah menjalankan simulasi dengan menggunakan aplikasi MATLAB, dapat disimpulkan bahwa pendeteksian objek bergerak dengan *Background Subtraction* berbasis *Gaussian Mixture Model* dipengaruhi oleh beberapa parameter yang akan mempengaruhi hasil pemrosesannya, yaitu jumlah *gaussian mixture*, *learning rate* dan porsi minimum *background*. Parameter tersebut dapat diubah sesuai kebutuhan dan keinginan dari *user*. Simulasi ini dapat dijalankan dengan mudah dan tampilannya cukup *user friendly* sehingga *user* yang masih awam dengan simulasi pendeteksian objek bergerak tetap dapat menggunakan dan mempelajarinya. Namun, tampilan aplikasi utama dari program ini juga dapat dikembangkan.

Diketahui bahwa ketika jumlah GMM meningkat maka akan meningkatkan waktu komputasi, dimana akan lebih banyak distribusi gaussian yang perlu diperhitungkan pada observasi piksel tertentu. Lalu, didapat nilai *learning rate* efektif pada simulasi ini sebesar 0,00005. Diketahui juga bahwa nilai porsi minimum background memiliki dampak visual yang signifikan terhadap hasil deteksi. Dampak parameter lainnya seperti "Jumlah frame untuk *train*" juga dapat dipelajari lebih lanjut dengan menjalankan berbagai variasi simulasi lainnya.

REFERENSI

- C. Stauffer, W.E.L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking," in *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, 1999, doi: 10.1109/CVPR.1999.784637.
- Kaewtrakulpong, P. and R. Bowden. "An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection." in *Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01, VIDEO BASED SURVEILLANCE SYSTEMS: Computer Vision and Distributed Processing*, 2001.
- "quarter DIP Gaussian Mixture Models for Background Subtraction." [Online]. Available: https://youtu.be/g_ve2txPkSc. [Accessed: 23-May-2021].
- "vision.ForegroundDetector." [Online]. Available: https://www.mathworks.com/help/vision/ref/vision.foregrounddetector-system-object.html. [Accessed: 13-May-2021].
- Deisenroth, M.P., Faisal, A.A., Ong, C.S. (2020) Mathematics for Machine Learning. (Cambridge University Press, Cambridge).
- Efford, N. (2000) Digital Image Processing: A Practical Introduction Using JavaTM. (Pearson Education, Hoboken)
- "Morphological Image Processing." [Online]. Available: https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm. [Accessed: 23-May-2021].
- "Morphological Operations in Image Processing." [Online]. Available: https://himnickson.medium.com/morphological-operations-in-image-processing-cb8045b98fcc. [Accessed: 23-May-2021].
- "Tracking Moving Objects Using Kalman Filter." [Online]. Available: https://youtu.be/iFNhkw4oe7A. [Accessed: 23-May-2021].
- "How to Pick the Best Learning Rate for Your Machine Learning Project." [Online]. Available:https://medium.com/octavian-ai/which-optimizer-and-learning-rate-should-i-use-for-deep-learning-5acb418f9b2. [Accessed: 23-May-2021].

LAMPIRAN

1. Kode Program “DOBmGMM.m”

```
%% --KELOMPOK 6-------------------------
% Elyaser Ben Guno (1806195135)
% Thara Adiva Putri Candra (1806187594)
%% -------------------------------------
%parameter yang dimasukkan oleh user
filename = evalin('base','filename');
numGauss = evalin('base','numGauss');
numTFrames = evalin('base','numTFrames');
minBGRatio = evalin('base','minBGRatio');
learnRate = evalin('base','learnRate');

% Membaca video
reader = VideoReader(filename);

% ForegroundDetector mengimplementasikan gaussian mixture model
% untuk mensegmentasi objek bergerak. outputnya adalah binary mask
detector = vision.ForegroundDetector('NumGaussians', numGauss, ...
    'NumTrainingFrames', numTFrames, 'MinimumBackgroundRatio', minBGRatio,
...
    'LearningRate', learnRate);

% Grup piksel foreground yang terkoneksi akan dideteksi sebagai blob
% oleh blob detection. lalu dihitung area, centroid dan bounding boxnya
blobAnalyser = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...
    'AreaOutputPort', true, 'CentroidOutputPort', true, ...
    'MinimumBlobArea', 400);
% -------------------------------------
tracks = struct(...
    'id', {}, ...
    'bbox', {}, ...
    'kalmanFilter', {}, ...
    'age', {}, ...
    'totalVisibleCount', {}, ...
    'consecutiveInvisibleCount', {});
% -------------------------------------
nextId = 1;
numFrames = 0;

% Membuat objek untuk menyimpan video
v = VideoWriter('frame.avi');
w = VideoWriter('mask.avi');
afGMM = VideoWriter('afterGMM.avi');
open(v); open(w); open(afGMM);

%% -------------------------------------
% Deteksi Objek
% -------------------------------------
while hasFrame(reader)
    frame = readFrame(reader);
    % Deteksi foreground.
    mask = detector.step(frame);
    afterGMM = mask;
    % Mengaplikasikan operasi morfologi untuk hilangkan noise
```

```
% dan mengisi lubang-lubang
mask = imopen(mask, strel('rectangle', [3,3]));
mask = imclose(mask, strel('rectangle', [15, 15]));
mask = imfill(mask, 'holes');

% Melakukan analisis blob untuk mencari komponen terhubung.
[~, centroids, bboxes] = blobAnalyser.step(mask);

%% ----------------------------------------
% Prediksi lokasi baru dari track
% ----------------------------------------
for i = 1:length(tracks)
    bbox = tracks(i).bbox;

    % Prediksi lokasi track saat ini menggunakan kalman filter
    predictedCentroid = predict(tracks(i).kalmanFilter);

    % Menggeser bounding box agar tengahnya berada di
    % lokasi yang diprediksi
    predictedCentroid = int32(predictedCentroid) - bbox(3:4) / 2;
    tracks(i).bbox = [predictedCentroid, bbox(3:4)];
end
%% ----------------------------------------
% Menetapkan deteksi pada track
% ----------------------------------------
nTracks = length(tracks);
nDetections = size(centroids, 1);

% Menghitung cost dari penetapan tiap deteksi pada track
cost = zeros(nTracks, nDetections);
for i = 1:nTracks
    cost(i, :) = distance(tracks(i).kalmanFilter, centroids);
end

% Menyelesaikan permasalahan assignment
costOfNonAssignment = 20;
[assignments, unassignedTracks, unassignedDetections] = ...
    assignDetectionsToTracks(cost, costOfNonAssignment);

%% ----------------------------------------
% memperbarui assigned track
% ----------------------------------------
numAssignedTracks = size(assignments, 1);
for i = 1:numAssignedTracks
    trackIdx = assignments(i, 1);
    detectionIdx = assignments(i, 2);
    centroid = centroids(detectionIdx, :);
    bbox = bboxes(detectionIdx, :);

    % Mengoreksi estimasi dari lokasi objek
    % menggunakan deteksi baru
    correct(tracks(trackIdx).kalmanFilter, centroid);

    % Mengganti bounding box yang telah diprediksi
    % dengan bounding box terdeteksi
    tracks(trackIdx).bbox = bbox;

    % Memperbarui track's age.
    tracks(trackIdx).age = tracks(trackIdx).age + 1;
```

```
    % Memperbarui visibilitas
    tracks(trackIdx).totalVisibleCount = ...
    tracks(trackIdx).totalVisibleCount + 1;
    tracks(trackIdx).consecutiveInvisibleCount = 0;
end

%% ---------------------------------------
% Memperbarui unassigned tracks
% ---------------------------------------
for i = 1:length(unassignedTracks)
    ind = unassignedTracks(i);
    tracks(ind).age = tracks(ind).age + 1;
    tracks(ind).consecutiveInvisibleCount = ...
        tracks(ind).consecutiveInvisibleCount + 1;
end

%% ---------------------------------------
% Menghapus track yang telah hilang
% ---------------------------------------
invisibleForTooLong = 20;
ageThreshold = 8;

% Menghitung fraksi dari track's age
ages = [tracks(:).age];
totalVisibleCounts = [tracks(:).totalVisibleCount];
visibility = totalVisibleCounts ./ ages;

% Menemukan indeks dari track yang hilang
lostInds = (ages < ageThreshold & visibility < 0.6) | ...
    [tracks(:).consecutiveInvisibleCount] >= invisibleForTooLong;

% Menghapus track yang hilang
tracks = tracks(~lostInds);

%% ---------------------------------------
% Membuat track baru
% ---------------------------------------
centroids = centroids(unassignedDetections, :);
bboxes = bboxes(unassignedDetections, :);

for i = 1:size(centroids, 1)
    centroid = centroids(i,:);
    bbox = bboxes(i, :);

    % Membuat objek kalman filter
    kalmanFilter = configureKalmanFilter('ConstantVelocity', ...
        centroid, [200, 50], [100, 25], 100);

    % Membuat track baru
    newTrack = struct(...
        'id', nextId, ...
        'bbox', bbox, ...
        'kalmanFilter', kalmanFilter, ...
        'age', 1, ...
        'totalVisibleCount', 1, ...
        'consecutiveInvisibleCount', 0);

    % Menambahkan track baru ke array track
```

```
            tracks(end + 1) = newTrack;

            % Inkremen next id
            nextId = nextId + 1;
        end
        %% ------------------------------------------------
        % Menambahkan hasil tracking & deteksi ke frame & mask
        % ------------------------------------------------
        % Convert frame dan mask ke uint8 RGB.
        frame = im2uint8(frame);
        mask = uint8(repmat(mask, [1, 1, 3])) .* 255;
        afterGMM = uint8(repmat(afterGMM, [1, 1, 3])) .* 255;

        minVisibleCount = 8;
        if ~isempty(tracks)

            % Noisy detections tend to result in short-lived tracks.
            % Only display tracks that have been visible for more than
            % a minimum number of frames.
            reliableTrackInds = ...
                [tracks(:).totalVisibleCount] > minVisibleCount;
            reliableTracks = tracks(reliableTrackInds);

            if ~isempty(reliableTracks)
                % Mendapatkan bounding box
                bboxes = cat(1, reliableTracks.bbox);

                % Mendapatkan id
                ids = int32([reliableTracks(:).id]);

                % Membuat label untuk objek
                labels = cellstr(int2str(ids'));
                predictedTrackInds = ...
                    [reliableTracks(:).consecutiveInvisibleCount] > 0;
                isPredicted = cell(size(labels));
                isPredicted(predictedTrackInds) = {' predicted'};
                labels = strcat(labels, isPredicted);

                % Menambahkan objek ke frame
                frame = insertObjectAnnotation(frame, 'rectangle', ...
                    bboxes, labels);

                % Menambahkan objek ke mask
                mask = insertObjectAnnotation(mask, 'rectangle', ...
                    bboxes, labels);
            end
        end

        numFrames = numFrames + 1;
        % Display the mask and the frame.
        writeVideo(v, frame);
        writeVideo(w, mask);
        writeVideo(afGMM, afterGMM);
    end

    assignin('base','numFrames',numFrames);
    close(v); close(w); close(afGMM);
```

2. Kode Aplikasi “MotionObjectDetection.mlapp”

```
classdef MotionObjectDetector < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel  matlab.ui.container.Panel
        DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel  matlab.ui.control.Label
        ElyaserBenGuno1806195135ThparticaAdivaPutriCandra1806187594Label  matlab.ui.control.Label
        ParameterPanel                  matlab.ui.container.Panel
        Numberofgaussianmodesinthemixturemodel345EditFieldLabel  matlab.ui.control.Label
        Numberofgaussianmodesinthemixturemodel345EditField  matlab.ui.control.NumericEditField
        NumberoftrainingframesEditFieldLabel  matlab.ui.control.Label
        NumberoftrainingframesEditField  matlab.ui.control.NumericEditField
        MinimumbackgroundratioEditFieldLabel  matlab.ui.control.Label
        MinimumbackgroundratioEditField  matlab.ui.control.NumericEditField
        LearningRateEditFieldLabel      matlab.ui.control.Label
        LearningRateEditField           matlab.ui.control.NumericEditField
        UploadVideoButton               matlab.ui.control.Button
        VideoteruploadLabel             matlab.ui.control.Label
        EditField                       matlab.ui.control.EditField
        Label                           matlab.ui.control.Label
        Label_2                         matlab.ui.control.Label
        default3Label                   matlab.ui.control.Label
        default07Label                  matlab.ui.control.Label
        default40Label                  matlab.ui.control.Label
        default0005Label                matlab.ui.control.Label
        acceptedformataviLabel          matlab.ui.control.Label
        ProsesPanel                     matlab.ui.container.Panel
        ProsesButton                    matlab.ui.control.Button
        WaktuprosessekonLabel           matlab.ui.control.Label
        WaktuprosessekonEditField       matlab.ui.control.EditField
        PemutarVideoPanel               matlab.ui.container.Panel
        PlayButton_2                    matlab.ui.control.Button
        PlayButton                      matlab.ui.control.Button
        PlayButton_3                    matlab.ui.control.Button
        HasilGMMMaskingLabel            matlab.ui.control.Label
        MaskingHasilDeteksiLabel        matlab.ui.control.Label
        HasilGMMHasilDeteksiLabel       matlab.ui.control.Label
        UniversitasIndonesiaLabel       matlab.ui.control.Label
        FrameSelectorPanel              matlab.ui.container.Panel
        JumlahFrameEditFieldLabel       matlab.ui.control.Label
        JumlahFrameEditField            matlab.ui.control.EditField
        FrameyangingindilihatEditFieldLabel  matlab.ui.control.Label
        FrameyangingindilihatEditField  matlab.ui.control.NumericEditField
        TampilkanButton                 matlab.ui.control.Button
        HelpButton                      matlab.ui.control.Button
    end

    methods (Access = public)

    end
```

```
methods (Access = private)
    % Value changed function:
    % Numberofgaussianmodesinthemixturemodel345EditField
    function Numberofgaussianmodesinthemixturemodel345EditFieldValueChanged(app, event)
        numGauss = app.Numberofgaussianmodesinthemixturemodel345EditField.Value;
        assignin('base','numGauss',numGauss');
    end
    % Value changed function: NumberoftrainingframesEditField
    function NumberoftrainingframesEditFieldValueChanged(app, event)
        numTFrames = app.NumberoftrainingframesEditField.Value;
        assignin('base','numTFrames',numTFrames);
    end
    % Value changed function: MinimumbackgroundratioEditField
    function MinimumbackgroundratioEditFieldValueChanged(app, event)
        minBGRatio = app.MinimumbackgroundratioEditField.Value;
        assignin('base','minBGRatio',minBGRatio);
    end
    % Value changed function: LearningRateEditField
    function LearningRateEditFieldValueChanged(app, event)
        learnRate = app.LearningRateEditField.Value;
        assignin('base','learnRate',learnRate);
    end
    % Button pushed function: ProsesButton
    function ProsesButtonPushed(app, event)
        app.WaktuprosessekonEditField.Value = 'processing...';
        tic
        run('DOBmGMM.m');
        waktuProses = toc
        waktuProsesText = string(waktuProses);
        app.WaktuprosessekonEditField.Value = waktuProsesText;
        numFrames = evalin('base','numFrames');
        app.JumlahFrameEditField.Value = string(numFrames);
    end
    % Button pushed function: PlayButton_2
    function PlayButton_2Pushed2(app, event)
        numFrames = evalin('base','numFrames');
        reader1 = VideoReader('frame.avi');
        reader2 = VideoReader('mask.avi');
        reader3 = VideoReader('afterGMM.avi');

        videoPlayer = vision.VideoPlayer('Position', [700, 200, 700, 400]);
        maskPlayer = vision.VideoPlayer('Position', [0, 200, 700, 400]);

        for i=1:1:numFrames-1
            frame = readFrame(reader1);
            mask = readFrame(reader2);

            maskPlayer.step(mask);
            videoPlayer.step(frame);
            pause(1/30);
        end
    end
```

```
% Button pushed function: UploadVideoButton
function UploadVideoButtonPushed(app, event)
    filename = uigetfile('*.avi');
    assignin('base','filename',filename);
    app.EditField.Value = filename;
end
% Button pushed function: PlayButton
function PlayButtonPushed(app, event)
    numFrames = evalin('base','numFrames');
    reader1 = VideoReader('frame.avi');
    reader2 = VideoReader('mask.avi');
    reader3 = VideoReader('afterGMM.avi');

    maskPlayer = vision.VideoPlayer('Position', [700, 200, 700, 400]);
    GMMPlayer = vision.VideoPlayer('Position', [0, 200, 700, 400]);

    for i=1:1:numFrames-1
        GMM = readFrame(reader3);
        mask = readFrame(reader2);

        GMMPlayer.step(GMM);
        maskPlayer.step(mask);
        pause(1/30);
    end
end
% Button pushed function: PlayButton_3
function PlayButton_3Pushed(app, event)
    numFrames = evalin('base','numFrames');
    reader1 = VideoReader('frame.avi');
    reader2 = VideoReader('mask.avi');
    reader3 = VideoReader('afterGMM.avi');

    videoPlayer = vision.VideoPlayer('Position', [700, 200, 700, 400]);
    GMMPlayer = vision.VideoPlayer('Position', [0, 200, 700, 400]);

    for i=1:1:numFrames-1
        GMM = readFrame(reader3);
        frame = readFrame(reader1);

        GMMPlayer.step(GMM);
        videoPlayer.step(frame);
        pause(1/30);
    end
end
% Button pushed function: TampilkanButton
function TampilkanButtonPushed(app, event)
    numfr = app.FrameyangingindilihatEditField.Value;
    numFrames = evalin('base','numFrames');

    if numfr > numFrames
        numfr = numFrames
    end
```

```
            asli = VideoReader('videotest1.avi');
            afGMM = VideoReader('afterGMM.avi');
            fr = VideoReader('frame.avi');
            ms = VideoReader('mask.avi');

            figure('units','normalized','outerposition',[0 0 1 1])
            subplot(2,2,1);
            imshow(read(asli,numfr)); title('video asli');
            imwrite(read(asli,numfr),'1. video asli.png');
            subplot(2,2,2);
            imshow(read(fr,numfr)); title('video hasil deteksi');
            imwrite(read(fr,numfr),'4. video hasil deteksi.png');
            subplot(2,2,3);
            imshow(read(afGMM,numfr)); title('deteksi GMM');
            imwrite(read(afGMM,numfr),'2. deteksi GMM.png');
            subplot(2,2,4);
            imshow(read(ms,numfr)); title('mask (deteksi GMM setelah operasi morfologi');
            imwrite(read(ms,numfr),'3. setelah morfologi.png');
        end
        % Button pushed function: HelpButton
        function HelpButtonPushed(app, event)
            f = msgbox({'petunjuk penggunaan:',...
                '1. upload video yang ingin diproses (format: *.avi)',...
                '2. masukkan parameter inputnya:',...
                '   jumlah GMM, defaultnya 3, 4 atau 5',...
                '   jumlah training frames, defaultnya 40',...
                '   porsi minimum background, rentangnya 0-1, defaultnya 0,7',...
                '   learning rate, rentangnya 0-1, defaultnya 0,005',...
                '3. lalu, tekan tombol proses, tunggu',...
                '4. setelah selesai, akan muncul waktu komputasi dan',...
                '   jumlah frame yang ada',...
                '5. user bisa menampilkan video hasil proses,',...
                '   hanya dua video yang bisa ditampilkan dalam ',...
                '   satu layar',...
                '6. user juga dapat meninjau frame tertentu',...
                '   melalui frame selector'});
            set(f, 'position', [300 100 350 300]); %makes box bigger
            ah = get( f, 'CurrentAxes' );
            ch = get( ah, 'Children' );
            set( ch, 'FontSize', 12 ); %makes text bigger
        end
    end
    % App initialization and construction
    methods (Access = private)
        % Create UIFigure and components
        function createComponents(app)
            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Color = [0.4353 0.6353 0.9294];
            app.UIFigure.Position = [100 100 720 538];
            app.UIFigure.Name = 'UI Figure';
            % Create ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel
            app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel = uipanel(app.UIFigure);
```

```
            app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel.Title = 'Proyek akhir kelas
pengolahan sinyal dan layanan multimedia  ';
            app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel.BackgroundColor = [0.9412
0.9412 0.9412];
            app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel.FontSize = 16;
            app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel.Position = [23 393 685 125];
            % Create DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel =
uilabel(app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel);
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel.FontName = 'Arial';
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel.FontSize = 28;
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel.FontWeight = 'bold';
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel.Position = [11 32 673 64];
            app.DeteksiObjekBergerakMenggunakanGaussianMixtureModelLabel.Text = {'Deteksi Objek
Bergerak Menggunakan Gaussian '; 'Mixture Model'};
            % Create ElyaserBenGuno1806195135ThataAdivaPutriCandra1806187594Label
            app.ElyaserBenGuno1806195135ThataAdivaPutriCandra1806187594Label =
uilabel(app.ProyekakhirkelaspengolahansinyaldanlayananmultimediaPanel);
            app.ElyaserBenGuno1806195135ThataAdivaPutriCandra1806187594Label.FontSize = 16;
            app.ElyaserBenGuno1806195135ThataAdivaPutriCandra1806187594Label.Position = [11 8 555 22];
            app.ElyaserBenGuno1806195135ThataAdivaPutriCandra1806187594Label.Text = 'Elyaser Ben Guno
(1806195135) & Thara Adiva Putri Candra (1806187594)';
            % Create ParameterPanel
            app.ParameterPanel = uipanel(app.UIFigure);
            app.ParameterPanel.Title = 'Parameter';
            app.ParameterPanel.BackgroundColor = [0.9412 0.9412 0.9412];
            app.ParameterPanel.FontWeight = 'bold';
            app.ParameterPanel.FontSize = 14;
            app.ParameterPanel.Position = [23 214 685 170];
            % Create Numberofgaussianmodesinthemixturemodel345EditFieldLabel
            app.Numberofgaussianmodesinthemixturemodel345EditFieldLabel = uilabel(app.ParameterPanel);
            app.Numberofgaussianmodesinthemixturemodel345EditFieldLabel.FontSize = 14;
            app.Numberofgaussianmodesinthemixturemodel345EditFieldLabel.Position = [190 84 135 48];
            app.Numberofgaussianmodesinthemixturemodel345EditFieldLabel.Text = {'Number of gaussian';
'modes in the mixture'; 'model (3 / 4 / 5)'};
            % Create Numberofgaussianmodesinthemixturemodel345EditField
            app.Numberofgaussianmodesinthemixturemodel345EditField = uieditfield(app.ParameterPanel,
'numeric');
            app.Numberofgaussianmodesinthemixturemodel345EditField.ValueChangedFcn =
createCallbackFcn(app, @Numberofgaussianmodesinthemixturemodel345EditFieldValueChanged, true);
            app.Numberofgaussianmodesinthemixturemodel345EditField.FontSize = 14;
            app.Numberofgaussianmodesinthemixturemodel345EditField.Position = [335 100 83 22];
            % Create NumberoftrainingframesEditFieldLabel
            app.NumberoftrainingframesEditFieldLabel = uilabel(app.ParameterPanel);
            app.NumberoftrainingframesEditFieldLabel.FontSize = 14;
            app.NumberoftrainingframesEditFieldLabel.Position = [463 84 71 48];
            app.NumberoftrainingframesEditFieldLabel.Text = {'Number of'; 'training'; 'frames'};
            % Create NumberoftrainingframesEditField
            app.NumberoftrainingframesEditField = uieditfield(app.ParameterPanel, 'numeric');
            app.NumberoftrainingframesEditField.ValueChangedFcn = createCallbackFcn(app,
@NumberoftrainingframesEditFieldValueChanged, true);
            app.NumberoftrainingframesEditField.FontSize = 14;
            app.NumberoftrainingframesEditField.Position = [562 100 91 22];
```

```
            % Create MinimumbackgroundratioEditFieldLabel
            app.MinimumbackgroundratioEditFieldLabel = uilabel(app.ParameterPanel);
            app.MinimumbackgroundratioEditFieldLabel.FontSize = 14;
            app.MinimumbackgroundratioEditFieldLabel.Position = [190 24 79 48];
            app.MinimumbackgroundratioEditFieldLabel.Text = {'Minimum'; 'background'; 'ratio'};
            % Create MinimumbackgroundratioEditField
            app.MinimumbackgroundratioEditField = uieditfield(app.ParameterPanel, 'numeric');
            app.MinimumbackgroundratioEditField.ValueChangedFcn = createCallbackFcn(app,
@MinimumbackgroundratioEditFieldValueChanged, true);
            app.MinimumbackgroundratioEditField.FontSize = 14;
            app.MinimumbackgroundratioEditField.Position = [334 39 85 22];
            % Create LearningRateEditFieldLabel
            app.LearningRateEditFieldLabel = uilabel(app.ParameterPanel);
            app.LearningRateEditFieldLabel.FontSize = 14;
            app.LearningRateEditFieldLabel.Position = [463 32 60 32];
            app.LearningRateEditFieldLabel.Text = {'Learning'; 'Rate'};
            % Create LearningRateEditField
            app.LearningRateEditField = uieditfield(app.ParameterPanel, 'numeric');
            app.LearningRateEditField.ValueChangedFcn = createCallbackFcn(app,
@LearningRateEditFieldValueChanged, true);
            app.LearningRateEditField.FontSize = 14;
            app.LearningRateEditField.Position = [562 39 91 22];
            % Create UploadVideoButton
            app.UploadVideoButton = uibutton(app.ParameterPanel, 'push');
            app.UploadVideoButton.ButtonPushedFcn = createCallbackFcn(app, @UploadVideoButtonPushed,
true);
            app.UploadVideoButton.FontSize = 14;
            app.UploadVideoButton.Position = [30 99 110 24];
            app.UploadVideoButton.Text = 'Upload Video';
            % Create VideoteruploadLabel
            app.VideoteruploadLabel = uilabel(app.ParameterPanel);
            app.VideoteruploadLabel.FontSize = 14;
            app.VideoteruploadLabel.Position = [33 50 107 22];
            app.VideoteruploadLabel.Text = 'Video terupload:';
            % Create EditField
            app.EditField = uieditfield(app.ParameterPanel, 'text');
            app.EditField.FontSize = 14;
            app.EditField.Position = [30 26 110 22];
            % Create Label
            app.Label = uilabel(app.ParameterPanel);
            app.Label.FontSize = 14;
            app.Label.Position = [222 21 43 22];
            app.Label.Text = '(0 - 1)';
            % Create Label_2
            app.Label_2 = uilabel(app.ParameterPanel);
            app.Label_2.FontSize = 14;
            app.Label_2.Position = [497 29 43 22];
            app.Label_2.Text = '(0 - 1)';
            % Create default3Label
            app.default3Label = uilabel(app.ParameterPanel);
            app.default3Label.FontColor = [0.502 0.502 0.502];
            app.default3Label.Position = [335 79 55 22];
            app.default3Label.Text = 'default: 3';
```

```
% Create default07Label
app.default07Label = uilabel(app.ParameterPanel);
app.default07Label.FontColor = [0.502 0.502 0.502];
app.default07Label.Position = [334 18 65 22];
app.default07Label.Text = 'default: 0,7';
% Create default40Label
app.default40Label = uilabel(app.ParameterPanel);
app.default40Label.FontColor = [0.502 0.502 0.502];
app.default40Label.Position = [562 79 62 22];
app.default40Label.Text = 'default: 40';
% Create default0005Label
app.default0005Label = uilabel(app.ParameterPanel);
app.default0005Label.FontColor = [0.502 0.502 0.502];
app.default0005Label.Position = [562 18 78 22];
app.default0005Label.Text = 'default: 0,005';
% Create acceptedformataviLabel
app.acceptedformataviLabel = uilabel(app.ParameterPanel);
app.acceptedformataviLabel.FontColor = [0.502 0.502 0.502];
app.acceptedformataviLabel.Position = [30 79 122 22];
app.acceptedformataviLabel.Text = 'accepted format: *.avi';
% Create ProsesPanel
app.ProsesPanel = uipanel(app.UIFigure);
app.ProsesPanel.Title = 'Proses';
app.ProsesPanel.FontWeight = 'bold';
app.ProsesPanel.FontSize = 14;
app.ProsesPanel.Position = [23 44 185 163];
% Create ProsesButton
app.ProsesButton = uibutton(app.ProsesPanel, 'push');
app.ProsesButton.ButtonPushedFcn = createCallbackFcn(app, @ProsesButtonPushed, true);
app.ProsesButton.FontSize = 14;
app.ProsesButton.Position = [30 104 110 24];
app.ProsesButton.Text = 'Proses';
% Create WaktuprosessekonLabel
app.WaktuprosessekonLabel = uilabel(app.ProsesPanel);
app.WaktuprosessekonLabel.FontSize = 14;
app.WaktuprosessekonLabel.Position = [29 51 141 42];
app.WaktuprosessekonLabel.Text = 'Waktu proses (sekon)';
% Create WaktuprosessekonEditField
app.WaktuprosessekonEditField = uieditfield(app.ProsesPanel, 'text');
app.WaktuprosessekonEditField.FontSize = 14;
app.WaktuprosessekonEditField.Position = [29 35 111 22];
% Create PemutarVideoPanel
app.PemutarVideoPanel = uipanel(app.UIFigure);
app.PemutarVideoPanel.Title = 'Pemutar Video';
app.PemutarVideoPanel.FontWeight = 'bold';
app.PemutarVideoPanel.FontSize = 14;
app.PemutarVideoPanel.Position = [219 44 248 163];
% Create PlayButton_2
app.PlayButton_2 = uibutton(app.PemutarVideoPanel, 'push');
app.PlayButton_2.ButtonPushedFcn = createCallbackFcn(app, @PlayButton_2Pushed2, true);
app.PlayButton_2.FontSize = 14;
app.PlayButton_2.Position = [139 63 83 24];
app.PlayButton_2.Text = 'Play';
```

```
% Create PlayButton
app.PlayButton = uibutton(app.PemutarVideoPanel, 'push');
app.PlayButton.ButtonPushedFcn = createCallbackFcn(app, @PlayButtonPushed, true);
app.PlayButton.FontSize = 14;
app.PlayButton.Position = [139 103 83 24];
app.PlayButton.Text = 'Play';
% Create PlayButton_3
app.PlayButton_3 = uibutton(app.PemutarVideoPanel, 'push');
app.PlayButton_3.ButtonPushedFcn = createCallbackFcn(app, @PlayButton_3Pushed, true);
app.PlayButton_3.FontSize = 14;
app.PlayButton_3.Position = [137 22 85 24];
app.PlayButton_3.Text = 'Play';
% Create HasilGMMMaskingLabel
app.HasilGMMMaskingLabel = uilabel(app.PemutarVideoPanel);
app.HasilGMMMaskingLabel.FontSize = 14;
app.HasilGMMMaskingLabel.Position = [18 99 88 32];
app.HasilGMMMaskingLabel.Text = {'Hasil GMM &'; 'Masking'};
% Create MaskingHasilDeteksiLabel
app.MaskingHasilDeteksiLabel = uilabel(app.PemutarVideoPanel);
app.MaskingHasilDeteksiLabel.FontSize = 14;
app.MaskingHasilDeteksiLabel.Position = [18 59 87 32];
app.MaskingHasilDeteksiLabel.Text = {'Masking &'; 'Hasil Deteksi'};
% Create HasilGMMHasilDeteksiLabel
app.HasilGMMHasilDeteksiLabel = uilabel(app.PemutarVideoPanel);
app.HasilGMMHasilDeteksiLabel.FontSize = 14;
app.HasilGMMHasilDeteksiLabel.Position = [18 18 88 32];
app.HasilGMMHasilDeteksiLabel.Text = {'Hasil GMM &'; 'Hasil Deteksi'};
% Create UniversitasIndonesiaLabel
app.UniversitasIndonesiaLabel = uilabel(app.UIFigure);
app.UniversitasIndonesiaLabel.FontSize = 16;
app.UniversitasIndonesiaLabel.FontWeight = 'bold';
app.UniversitasIndonesiaLabel.Position = [536 11 171 22];
app.UniversitasIndonesiaLabel.Text = 'Universitas Indonesia';
% Create FrameSelectorPanel
app.FrameSelectorPanel = uipanel(app.UIFigure);
app.FrameSelectorPanel.Title = 'Frame Selector';
app.FrameSelectorPanel.FontWeight = 'bold';
app.FrameSelectorPanel.FontSize = 14;
app.FrameSelectorPanel.Position = [477 44 230 163];
% Create JumlahFrameEditFieldLabel
app.JumlahFrameEditFieldLabel = uilabel(app.FrameSelectorPanel);
app.JumlahFrameEditFieldLabel.HorizontalAlignment = 'right';
app.JumlahFrameEditFieldLabel.FontSize = 14;
app.JumlahFrameEditFieldLabel.Position = [23 105 95 22];
app.JumlahFrameEditFieldLabel.Text = 'Jumlah Frame';
% Create JumlahFrameEditField
app.JumlahFrameEditField = uieditfield(app.FrameSelectorPanel, 'text');
app.JumlahFrameEditField.FontSize = 14;
app.JumlahFrameEditField.Position = [137 105 62 22];
% Create FrameyangingindilihatEditFieldLabel
app.FrameyangingindilihatEditFieldLabel = uilabel(app.FrameSelectorPanel);
app.FrameyangingindilihatEditFieldLabel.FontSize = 14;
app.FrameyangingindilihatEditFieldLabel.Position = [29 60 80 32];
```

```
            app.FrameyangingindilihatEditFieldLabel.Text = {'Frame yang'; 'ingin dilihat'};
            % Create FrameyangingindilihatEditField
            app.FrameyangingindilihatEditField = uieditfield(app.FrameSelectorPanel, 'numeric');
            app.FrameyangingindilihatEditField.FontSize = 14;
            app.FrameyangingindilihatEditField.Position = [137 70 62 22];
            % Create TampilkanButton
            app.TampilkanButton = uibutton(app.FrameSelectorPanel, 'push');
            app.TampilkanButton.ButtonPushedFcn = createCallbackFcn(app, @TampilkanButtonPushed,
true);
            app.TampilkanButton.FontSize = 14;
            app.TampilkanButton.Position = [64 22 100 24];
            app.TampilkanButton.Text = 'Tampilkan';
            % Create HelpButton
            app.HelpButton = uibutton(app.UIFigure, 'push');
            app.HelpButton.ButtonPushedFcn = createCallbackFcn(app, @HelpButtonPushed, true);
            app.HelpButton.FontSize = 14;
            app.HelpButton.Position = [23 10 94 24];
            app.HelpButton.Text = 'Help';
        end
    end
    methods (Access = public)
        % Construct app
        function app = MotionObjectDetector
            % Create and configure components
            createComponents(app)
            % Register the app with App Designer
            registerApp(app, app.UIFigure)
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```